Remarks

Applicants respectfully request reconsideration of the present application in view of the foregoing amendments and the following remarks. Claims 1, 3-13, 15, 17-21, 23, 24, 26, 28, 31-34, 36, 38, and 39 are pending in the application. No claims have been allowed. Claims 1, 20, 31, and 36 are independent. Claim 20 has been amended.

Cited Art

The Action cites Gordon, U.S. Patent No. 6,560,774 (hereinafter "Gordon"); Inside Microsoft .NET IL Assembler (hereinafter "Microsoft-IL"); and Syme, U.S. Patent No. 7,346,901 (hereinafter "Syme").

Claim Rejections under 35 U.S.C. § 103(a)

The Action rejects claims 1, 3-13, 15, 17-21, 23, 24, 26, 28, and 31-34 under 35 U.S.C § 103(a) as unpatentable over Gordon in view of Microsoft-IL. The Action further rejects claims 36, 38, and 39 under 35 U.S.C § 103(a) as unpatentable over Microsoft-IL in view of Syme and Gordon. Applicants respectfully traverse the rejections.

Claims 1, 3-13, 15, and 17-19 are Allowable Over Gordon in View of Microsoft-IL

Claim 1 recites a method of representing type information for a typed intermediate language via objects of classes in a class hierarchy, wherein the class hierarchy comprises at least one class and a plurality of sub-classes for representing different type classifications, the method comprising (emphasis added):

> instantiating one or more objects of one or more of the sub-classes of the hierarchy, wherein the one or more sub-classes represent classifications of types for the typed intermediate language; and
>
> storing information in the one or more objects;
>
> wherein the typed intermediate language is capable of representing a plurality of different programming languages, and wherein the one or more objects represent type information for instructions in the typed intermediate language;
>
> wherein the classifications of types comprises a primitive type associated with a primitive type size, and wherein the primitive type size is settable to represent a constant size, the primitive type size is settable to represent a symbolic size, and the primitive type size is settable to represent an unknown size; and

wherein one of the sub-classes representing a primitive type represents an unknown type, wherein the unknown type can represent any type, and wherein a compiler drops type information by changing a known type to the unknown type during a stage of lowering.

Neither Gordon nor Microsoft-IL, whether viewed separately or in combination with each other, teach or suggest the above language of claim 1. For example, neither Gordon nor Microsoft-IL teach or suggest "wherein one of the sub-classes representing a primitive type represents an unknown type, wherein the unknown type can represent any type, and wherein a compiler drops type information by changing a known type to the unknown type during a stage of lowering" as recited in claim 1.

1. Gordon does not teach or suggest dropping type information during lowering

Gordon does not teach or suggest, "wherein a compiler drops type information by changing a known type to the unknown type during a stage of lowering" as recited by claim 1.

The Examiner argues that Gordon describes this language, citing to Gordon at col. 27, line 64 to col. 28, line 7. The Examiner states that Gordon recites "type information is deferred until JIT compilation." Action, page 5. Applicants respectfully point out that the Examiner's language does not reflect the disclosure of Gordon. In fact, Gordon is not describing deferring type information. Instead, Gordon describes deferring size of a known type. Specifically, Gordon describes "natural size types (I, R, U, O, and &)," where the size of these types can be deferred when the IL is generated, "These data types have a fixed but unknown size when the IL is generated at compile time." (emphasis added) Gordon, col. 27, lines 63-67. For example, the unsigned integer type "U" may be a one-byte unsigned integer (U1), a two-byte unsigned integer (U2), etc., but the size is not set until JIT compilation time, when the target architecture is known. Gordon, col. 27, line 67 to col. 28, line 2. Of course, because the size is not specified at compile time, field offsets and stack frame offsets will also not be known until JIT compilation time. Gordon, col. 28, lines 2-4.

Instead of deferring size for a known type as described by Gordon, claim 1 recites, "wherein a compiler drops type information by changing a known type to the unknown type during a stage of lowering." For example, the Application at page 10, lines 3-8, describes dropping type information for a pointer to an integer by replacing the "integer type" with the

"unknown type" during a stage of lowering.

Furthermore, as understood by Applicants, Microsoft-IL does not cure this deficiency.

2. Gordon does not teach or suggest an unknown type

Gordon does not teach or suggest, "wherein one of the sub-classes representing a primitive type represents an unknown type, wherein the unknown type can represent any type," as recited by claim 1.

The Examiner argues that Gordon describes this language, citing to Gordon at col. 27, line 64 to col. 28, line 7. Specifically, the Examiner cites to Gordon's description of field offsets and stack frame offsets being unknown at compile time. Gordon at col. 28, lines 2-4. Applicants respectfully disagree.

As discussed above, Gordon describes "natural size types (I, R, U, O, and &)," where the size of these types can be deferred when the IL is generated, until JIT compile time when the target architecture is known. Gordon, col. 27, lines 63 to col. 28, line 2. Of course, because the size is not specified at compile time, field offsets and stack frame offsets will also not be known until JIT compilation time. Gordon, col. 28, lines 2-4.

Gordon's description is clear. Sizes can be deferred at IL/compilation time until JIT compile time (when the target architecture is known). Because of this, offsets may not be known (e.g., two-byte unsigned integers will have a different offset than four-byte unsigned integers). None of this description in Gordon describes "an unknown type, wherein the unknown type can represent any," as recited by claim 1. At most, Gordon describes an unknown size for a known type.

Furthermore, as understood by Applicants, Microsoft-IL does not cure this deficiency.

For at least the above reasons, Gordon and Microsoft-IL, whether considered separately or in combination with each other, do not teach or suggest each and every element of claim 1. Therefore, claim 1 should be in condition for allowance.

Dependent claims 3-13, 15, and 17-19 should be allowable for at least the reasons discussed above with regard to claim 1.

Claims 20, 21, 23-24, 26, and 28 are Allowable Over Gordon in View of Microsoft-IL

Independent claim 20 recites a computer-readable medium storing a software program thereon, the program comprising computer executable instructions for implementing a method for representing type information for a typed intermediate language using a class hierarchy for representing different type classifications, the method comprising (emphasis added):

> defining a programming class of the class hierarchy as 'PrimType', wherein the programming class represents primitive type information for the typed intermediate language;
>
> associating a size with instances of the 'PrimType' class, wherein the size is settable to represent an actual size of instances of the 'PrimType' class, settable to represent a symbolic size of instances of the 'PrimType' class, and settable to represent an unknown size of instances of the 'PrimType' class, and *wherein the actual size and the symbolic size are defined as a number of bits*; and
>
> associating a kind of type with instances of the 'PrimType' class;
>
> *wherein the class 'PrimType' represents a plurality of types comprising at least an unknown type, wherein the unknown type can represent any type, and wherein a compiler drops type information during compilation by changing a known type to the unknown type, in the typed intermediate language, during a stage of lowering.*

Regarding the amendments to claim 20, see the Application at, for example, page 5, lines 22-26 and page 9, line 20 to page 10, line 25.

For at least the reasons discussed above with regard to claim 1, neither Gordon nor Microsoft-IL teach or suggest, "wherein the class 'PrimType' represents a plurality of types comprising at least an unknown type, wherein the unknown type can represent any type, and wherein a compiler drops type information during compilation by changing a known type to the unknown type, in the typed intermediate language, during a stage of lowering," as recited by claim 20.

In addition, neither Gordon nor Microsoft-IL teach or suggest, "associating a size with instances of the 'PrimType' class, wherein the size is settable to represent an actual size of instances of the 'PrimType' class, settable to represent a symbolic size of instances of the 'PrimType' class, and settable to represent an unknown size of instances of the 'PrimType' class, and *wherein the actual size and the symbolic size are defined as a number of bits*," as recited by claim 20. Regarding defining primitive type sizes as a number of bits, the Examiner cites to Gordon at col. 17, line 42 to col. 18, line 10 and Fig. 12 (Action, page 9, with regard to claim 22, which has been previously canceled). These sections of Gordon describe various type fields, but

do not teach or suggest this language.

Because Gordon and Microsoft-IL, whether considered separately or in combination with each other, do not teach or suggest each and every element of amended independent claim 20, claim 20 is allowable over Gordon and Microsoft-IL. Dependent claims 21, 23-24, 26, and 28 are allowable at least because they depend from allowable independent claim 20. Applicants respectfully request withdrawal of the § 103 rejections and allowance of claims 20-21, 23-24, 26, and 28.

Claims 31-34 are Allowable Over Gordon in View of Microsoft-IL

Independent claim 31 recites a method for representing type information for a typed intermediate language using a class hierarchy by programmatically defining a type representation, the method comprising (emphasis added):

> defining a base class of the class hierarchy;
>
> defining a plurality of classes hierarchically below the base class, wherein the plurality of classes represent type information for the typed intermediate language, and wherein the plurality of classes represent at least pointer types, container types and function types of a plurality of programming languages, and wherein the plurality of classes further comprise primitive types and the primitive types are associated with a primitive type size settable to represent a constant size, settable to represent a symbolic size, and settable to represent an unknown size;
>
> *wherein one of the primitive types represents an unknown type, wherein the unknown type can represent any type, and wherein a compiler drops type information by changing a known type to the unknown type during a stage of lowering.*

For at least the reasons discussed above with regard to claim 1, neither Gordon nor Microsoft-IL teach or suggest, "wherein one of the primitive types represents an unknown type, wherein the unknown type can represent any type, and wherein a compiler drops type information by changing a known type to the unknown type during a stage of lowering," as recited by claim 31.

Because Gordon and Microsoft-IL, whether considered separately or in combination with each other, do not teach or suggest each and every element of independent claim 31, claim 31 is allowable over Gordon and Microsoft-IL. Dependent claims 32-34 are allowable at least because they depend from allowable independent claim 31. Applicants respectfully request withdrawal of the § 103 rejections and allowance of claims 31-34.

Claims 36, 38, and 39 are Allowable Over Microsoft-IL in View of Syme and Gordon

Independent claim 36 recites a computer-readable medium storing a software program thereon, the program comprising computer executable instructions for implementing a method for representing type information for a typed intermediate language using a class hierarchy for representing different type classifications, the method comprising, in part (emphasis added):

> defining a programming class of the class hierarchy as 'PrimType', wherein an object of class 'PrimType' is a type representation for the typed intermediate language for primitive types in a section of code written in one of a plurality of programming languages;
>
> wherein the object of class 'PrimType' is associated with a size settable to represent a constant size for the object of class 'PrimType', settable to represent a symbolic size for the object of class 'PrimType', and settable to represent an unknown size for the object of class 'PrimType'; and
>
> *wherein the class 'PrimType' represents a plurality of types comprising at least an unknown type, wherein the unknown type can represent any type, and wherein a compiler drops type information by changing a known type to the unknown type during a stage of lowering.*

For at least the reasons discussed above with regard to claim 1, neither Gordon nor Microsoft-IL teach or suggest, "wherein the class 'PrimType' represents a plurality of types comprising at least an unknown type, wherein the unknown type can represent any type, and wherein a compiler drops type information by changing a known type to the unknown type during a stage of lowering," as recited by claim 36. Furthermore, as understood by Applicants, Syme and Gordon do not cure this deficiency.

Because Microsoft-IL, Gordon, and Syme, whether considered separately or in combination with each other, do not teach or suggest each and every element of independent claim 36, claim 36 is allowable. Dependent claims 38 and 39 are allowable at least because they depend from independent claim 36. Applicants respectfully request withdrawal of the § 103 rejections and allowance of claims 36, 38, and 39.

Interview Request

If the claims are not found by the Examiner to be allowable, the Examiner is requested to call the undersigned attorney to set up an interview to discuss this application.

Conclusion

The claims should be allowable. Such action is respectfully requested.

Respectfully submitted,

KLARQUIST SPARKMAN, LLP

One World Trade Center, Suite 1600
121 S.W. Salmon Street
Portland, Oregon 97204
Telephone: (503) 595-5300
Facsimile: (503) 595-5301

By /Cory A. Jones/
Cory A. Jones
Registration No. 55,307